I2C

I2C Flash

```
#include <stdio.h>
#include <linux/i2c.h>
#include <linux/i2c-dev.h>
#include <sys/stat.h>
#include <sys/types.h>
#include <fcntl.h>
#include <string.h>
#include <stdlib.h>

#define I2C_DEFAULT_TIMEOUT     1
#define I2C_DEFAULT_RETRY       3

/*
 * fd       :
 * timeout  :
 * retry    :
 */
//
int i2c_set(int fd, unsigned int timeout, unsigned int retry)
{
    if (fd == 0 )
        return -1;

    if (ioctl(fd, I2C_TIMEOUT, timeout ? timeout : I2C_DEFAULT_TIMEOUT) < 0)
        return -1;
    if (ioctl(fd, I2C_RETRIES, retry ? retry : I2C_DEFAULT_RETRY) < 0)
        return -1;

    return 0;
}
/*
 * fd   :
 * addr : i2c
 * reg  :
 * val  :
 *
 */
int i2c_byte_write(int fd, unsigned char addr, unsigned char reg, unsigned char val)
{
    int ret = 0;
    unsigned char outbuf[2];
    struct i2c_rdwr_ioctl_data packets;
    struct i2c_msg messages;

    packets.nmsgs = 1;
    packets.msgs = &messages;

    //
    messages.addr = addr;
    messages.flags = 0;
    messages.len = 2;       //2
    messages.buf = outbuf;
    outbuf[0] = reg;
    outbuf[1] = val;

    ret = ioctl(fd, I2C_RDWR, (unsigned long)&packets);   //
    if (ret < 0)
        ret = -1;

    return ret;
}

/*
 * fd   :
 * addr : i2c
 * reg  :
```

```
 * val  :
 * len  :
 *
 *        24c0281pagepagepage
 *        page,
 */
int i2c_nbytes_write(int fd, unsigned char addr, unsigned char reg, unsigned char *val, int len)
{
    int ret = 0;
    struct i2c_rdwr_ioctl_data packets;
    struct i2c_msg messages;
    int i;

    packets.nmsgs = 1;
    packets.msgs = &messages;

    //
    messages.addr = addr;
    messages.flags = 0;         //write
    messages.len = len + 1;     //
    //
    messages.buf = (unsigned char *)malloc(len+1);
    if (NULL == messages.buf)
    {
        ret = -1;
        goto err;
    }

    messages.buf[0] = reg;
    for (i = 0; i < len; i++)
    {
        messages.buf[1+i] = val[i];
    }

    ret = ioctl(fd, I2C_RDWR, (unsigned long)&packets);//
    if (ret < 0){
        printf("write error!\n");
        return -1;
    }

err:
    free(messages.buf);

    return ret;
}

/*
 * fd   :
 * addr : i2c
 * val  :
 *
 */
int i2c_byte_read(int fd, unsigned char addr, unsigned char *val)
{
    int ret = 0;
    struct i2c_rdwr_ioctl_data packets;
    struct i2c_msg messages;

    packets.nmsgs = 1;              //,1
    packets.msgs = &messages;

    //
    messages.addr = addr;           //i2c
    messages.flags = I2C_M_RD;      //
    messages.len = 1;               //
    messages.buf = val;             //val

    ret = ioctl (fd, I2C_RDWR, (unsigned long)&packets);  //
    if (ret < 0)
        ret = -1;
```

```
    return ret;
}

/*
 * fd   :
 * addr : i2c
 * reg  :
 * val  :
 * len  :
 *  eeprom
 */
int i2c_nbytes_read(int fd, unsigned char addr, unsigned char reg, unsigned char *val, int len)
{
    int ret = 0;
    unsigned char outbuf;
    struct i2c_rdwr_ioctl_data packets;
    struct i2c_msg messages[2];

    /* 2
     * ,,
     * 2
     */
    packets.nmsgs = 2;
    //
    messages[0].addr = addr;
    messages[0].flags = 0;              //write
    messages[0].len = 1;                //
    messages[0].buf = &outbuf;          //
    outbuf = reg;
    //
    messages[1].len = len;                          //
    messages[1].addr = addr;                        //
    messages[1].flags = I2C_M_RD;                   //read
    messages[1].buf = val;

    packets.msgs = messages;

    ret = ioctl(fd, I2C_RDWR, (unsigned long)&packets); //i2c,
    if (ret < 0)
        ret = -1;

    return ret;
}
```